



openHPI
Einführung in die Mathematik der Algorithmik
– Dijkstra Analyse I

Thomas Bläsius und Pascal Lenzner

Hasso-Plattner-Institut

Universität Potsdam

Initialisierung:

- nur Knoten H markiert; Distanz zum Startknoten $H = 0$
- alle Nachbarn X des Startknotens H sind Randknoten mit $dist(X, H) = \text{Länge}(X, H)$; Vorgänger ist H

Solange unmarkierter Randknoten existiert:

- X sei unmarkierter Randknoten mit geringster Distanz zu H
- entferne X aus der Menge der Randknoten
- mache für alle unmarkierten Nachbarn Y von X folgendes:
 - falls Y kein Randknoten:
 - füge Y in die Menge der Randknoten ein
 - $dist(Y, H) = \text{Länge}(Y, X) + dist(X, H)$; Vorgänger von Y ist X
 - sonst:
 - falls $dist(Y, H) > \text{Länge}(Y, X) + dist(X, H)$:
 - aktualisiere $dist(Y, H)$ zu $\text{Länge}(Y, X) + dist(X, H)$
 - X wird Vorgänger von Y
- markiere X

Initialisierung:

- nur Knoten H markiert; Distanz zum Startknoten $H = 0$

H markieren, alle
anderen Knoten auf
unmarkiert setzen

Distanz von H auf 0,
Distanz aller anderen
Knoten auf ∞ setzen

Pro Knoten Vorgängereintrag
anlegen (anfangs leer)

Initialisierung:

$\Theta(1)$ pro Knoten

- nur Knoten H markiert; Distanz zum Startknoten $H = 0$
- alle Nachbarn X des Startknotens H sind Randknoten mit $dist(X, H) = \text{Länge}(X, H)$; Vorgänger ist H

alle Nachbarn von H in Priority Queue einfügen

für alle Nachbarn von H Distanz setzen

für alle Nachbarn von H Vorgänger setzen

Insert

pro Nachbar:
 $\Theta(\text{cost}(\text{Insert}))$

pro Nachbar: $\Theta(1)$

Initialisierung:

$\Theta(1)$ pro Knoten
pro Nachbar:
 $\Theta(\text{cost}(\text{Insert}))$

- nur Knoten H markiert; Distanz zum Startknoten $H = 0$
- alle Nachbarn X des Startknotens H sind Randknoten mit $\text{dist}(X, H) = \text{Länge}(X, H)$; Vorgänger ist H

Solange unmarkierter Randknoten existiert:

- X sei unmarkierter Randknoten mit geringster Distanz zu H
- entferne X aus der Menge der Randknoten

ExtractMin

Initialisierung:

$\Theta(1)$ pro Knoten

- nur Knoten H markiert; Distanz zum Startknoten $H = 0$

pro Nachbar:

$\Theta(\text{cost}(\text{Insert}))$

- alle Nachbarn X des Startknotens H sind Randknoten mit $\text{dist}(X, H) = \text{Länge}(X, H)$; Vorgänger ist H

Solange unmarkierter Randknoten existiert:

$\Theta(\text{cost}(\text{ExtractMin}))$

- X sei unmarkierter Randknoten mit geringster Distanz zu H
- entferne X aus der Menge der Randknoten
- mache für alle unmarkierten Nachbarn Y von X folgendes:
 - falls Y kein Randknoten:
 - füge Y in die Menge der Randknoten ein
 - $\text{dist}(Y, H) = \text{Länge}(Y, X) + \text{dist}(X, H)$; Vorgänger von Y ist X

teste, ob Y nicht-leeren PQ-Verweis hat

$\Theta(1)$

Insert

$\Theta(\text{cost}(\text{Insert}))$

Initialisierung:

$\Theta(1)$ pro Knoten

- nur Knoten H markiert; Distanz zum Startknoten $H = 0$

pro Nachbar:

$\Theta(\text{cost}(\text{Insert}))$

- alle Nachbarn X des Startknotens H sind Randknoten mit $\text{dist}(X, H) = \text{Länge}(X, H)$; Vorgänger ist H

Solange unmarkierter Randknoten existiert:

$\Theta(\text{cost}(\text{ExtractMin}))$

- X sei unmarkierter Randknoten mit geringster Distanz zu H
- entferne X aus der Menge der Randknoten

- mache für alle unmarkierten Nachbarn Y von X folgendes:

$\Theta(1)$

- falls Y kein Randknoten:

$\Theta(\text{cost}(\text{Insert}))$

- füge Y in die Menge der Randknoten ein

$\Theta(1)$

- $\text{dist}(Y, H) = \text{Länge}(Y, X) + \text{dist}(X, H)$; Vorgänger von Y ist X

Test in $\Theta(1)$ per Direktzugriff auf Priorität von Y , $\text{Länge}(Y, X)$ und $\text{dist}(X, H)$

DecreaseKey

- sonst:

- falls $\text{dist}(Y, H) > \text{Länge}(Y, X) + \text{dist}(X, H)$:

- aktualisiere $\text{dist}(Y, H)$ zu $\text{Länge}(Y, X) + \text{dist}(X, H)$

- X wird Vorgänger von Y

Initialisierung:

$\Theta(1)$ pro Knoten

- nur Knoten H markiert; Distanz zum Startknoten $H = 0$

pro Nachbar:

$\Theta(\text{cost}(\text{Insert}))$

- alle Nachbarn X des Startknotens H sind Randknoten mit $\text{dist}(X, H) = \text{Länge}(X, H)$; Vorgänger ist H

Solange unmarkierter Randknoten existiert:

$\Theta(\text{cost}(\text{ExtractMin}))$

- X sei unmarkierter Randknoten mit geringster Distanz zu H
- entferne X aus der Menge der Randknoten

- mache für alle unmarkierten Nachbarn Y von X folgendes:

$\Theta(1)$

- falls Y kein Randknoten:

$\Theta(\text{cost}(\text{Insert}))$

- füge Y in die Menge der Randknoten ein

$\Theta(1)$

- $\text{dist}(Y, H) = \text{Länge}(Y, X) + \text{dist}(X, H)$; Vorgänger von Y ist X

- sonst:

$\Theta(1)$

- falls $\text{dist}(Y, H) > \text{Länge}(Y, X) + \text{dist}(X, H)$:

$\Theta(\text{cost}(\text{DecreaseKey}))$

- aktualisiere $\text{dist}(Y, H)$ zu $\text{Länge}(Y, X) + \text{dist}(X, H)$

$\Theta(1)$

- X wird Vorgänger von Y

$\Theta(1)$

- markiere X

G ist zusammenhängend

Wie oft werden die Blöcke ausgeführt?

Betrachten Graph G mit n Knoten und $m \geq n - 1$ Kanten

1 mal

Initialisierung:

- $\Theta(n)$ ■ nur Knoten H markiert; Distanz zum Startknoten $H = 0$
- pro Nachbar: $\Theta(\text{cost}(\text{Insert}))$ ■ alle Nachbarn X des Startknotens H sind Randknoten mit $\text{dist}(X, H) = \text{Länge}(X, H)$; Vorgänger ist H

Jeder Knoten wird ≤ 1 mal markiert

$\Theta(n)$ mal

Solange unmarkierter Randknoten existiert:

- $\Theta(\text{cost}(\text{ExtractMin}))$ ■ X sei unmarkierter Randknoten mit geringster Distanz zu H
- entferne X aus der Menge der Randknoten

Jeder Knoten wird nur 1 mal in PQ eingefügt

- mache für alle unmarkierten Nachbarn Y von X folgendes:

- $\Theta(1)$ ■ falls Y kein Randknoten:

nur $\Theta(n)$ mal

- $\Theta(\text{cost}(\text{Insert}))$ ■ füge Y in die Menge der Randknoten ein
- $\Theta(1)$ ■ $\text{dist}(Y, H) = \text{Länge}(Y, X) + \text{dist}(X, H)$; Vorgänger von Y ist X

Jede Kante wird ≤ 2 mal betrachtet

$\Theta(m)$ mal

- sonst:

- $\Theta(1)$ ■ falls $\text{dist}(Y, H) > \text{Länge}(Y, X) + \text{dist}(X, H)$:
- $\Theta(\text{cost}(\text{DecreaseKey}))$ ■ aktualisiere $\text{dist}(Y, H)$ zu $\text{Länge}(Y, X) + \text{dist}(X, H)$
- $\Theta(1)$ ■ X wird Vorgänger von Y

$\Theta(n)$ mal

- $\Theta(1)$ ■ markiere X

Gesamtkosten: $\Theta(n \cdot \text{cost}(\text{Insert})) + \Theta(n \cdot \text{cost}(\text{ExtractMin})) + \Theta(m \cdot \text{cost}(\text{DecreaseKey})) + \Theta(n) + \Theta(m)$

Gesamtkosten: $\Theta(n \cdot \text{cost}(\text{Insert})) + \Theta(n \cdot \text{cost}(\text{ExtractMin})) + \Theta(m \cdot \text{cost}(\text{DecreaseKey})) + \Theta(n) + \Theta(m)$

- haben die Kosten von Dijkstras Algorithmus analysiert
- Vorgehen:
 - schätze die asymptotischen Kosten der einzelnen Operationen für den schlimmstmöglichen Fall ab
 - ermittle, wie oft jede Operation im schlimmsten Fall ausgeführt wird
 - multipliziere die Anzahl der Ausführungen mit den Kosten pro Ausführung

Beobachtung:

Die Gesamtkosten von Dijkstras Algorithmus hängen von der konkreten Implementierung der Priority Queue ab.